

Sistema de Gestão da Base de Dados de uma Cadeia de Supermercados



Diogo Pereira, nº 44640
Eduardo Graça, nº 46794
João Vieira, nº 53332

Grupo 29
Turno Prático 4
Professor Matthias Knorr

Objectivos do SGBD Cadeia de Supermercados

O âmbito do nosso trabalho é desenvolver uma base de dados de uma Cadeia de Supermercados capaz de gerir a distribuição de recursos (Empregados, Viaturas, Produtos...), de armazenar um histórico de vendas de Produtos e de armazenar informações sobre os seus Clientes e Fornecedores.

O objectivo principal desta será, através das várias relações existentes no SGBD (Sistema de Gestão da Base de Dados), determinar os custos associados à gestão e operação da Cadeia de Supermercados, de modo a verificar se esta tem lucro.

Descrição do SGBD Cadeia de Supermercados e Decisões tomadas no Modelo ER

A Base de Dados da Cadeia de Supermercados guarda informações sobre Pessoas, nomeadamente sobre os Clientes da Cadeia de Supermercados e sobre os seus Empregados. A cada Pessoa está associado um NIF, nome, morada e telefone. Um Cliente tem ainda associado o seu endereço de email, enquanto que os Empregados têm por sua vez associado o seu cargo e salário. Uma Pessoa pode ser um Cliente, Empregado ou ambos.

A Cadeia de Supermercados tem dois tipos de Instalações - Armazéns e Supermercados. Nos Armazéns e Supermercados desta Cadeia de Supermercados trabalham vários tipos de empregados. Uma Instalação pode ser um Supermercado ou um Armazém, mas não os dois.

Uma Instalação tem um identificador único, bem como a sua morada e a sua capacidade máxima. Um Supermercado tem associado um nome, enquanto que um Armazém possui informação relativa ao seu custo de aluguer.

Um Empregado tem que trabalhar em (pelo menos) uma Instalação. Pode trabalhar em várias Instalações se o seu cargo assim o exigir (exemplo: supervisor).

Existem vários Fornecedores associados à Cadeia de Supermercados, que fornecem os Produtos necessários para o funcionamento de cada Supermercado. Um Fornecedor fornece pelo menos um Produto, que é caracterizado por um preço de venda, um identificador único de produto e uma capacidade que descreve o espaço que este Produto ocupa.

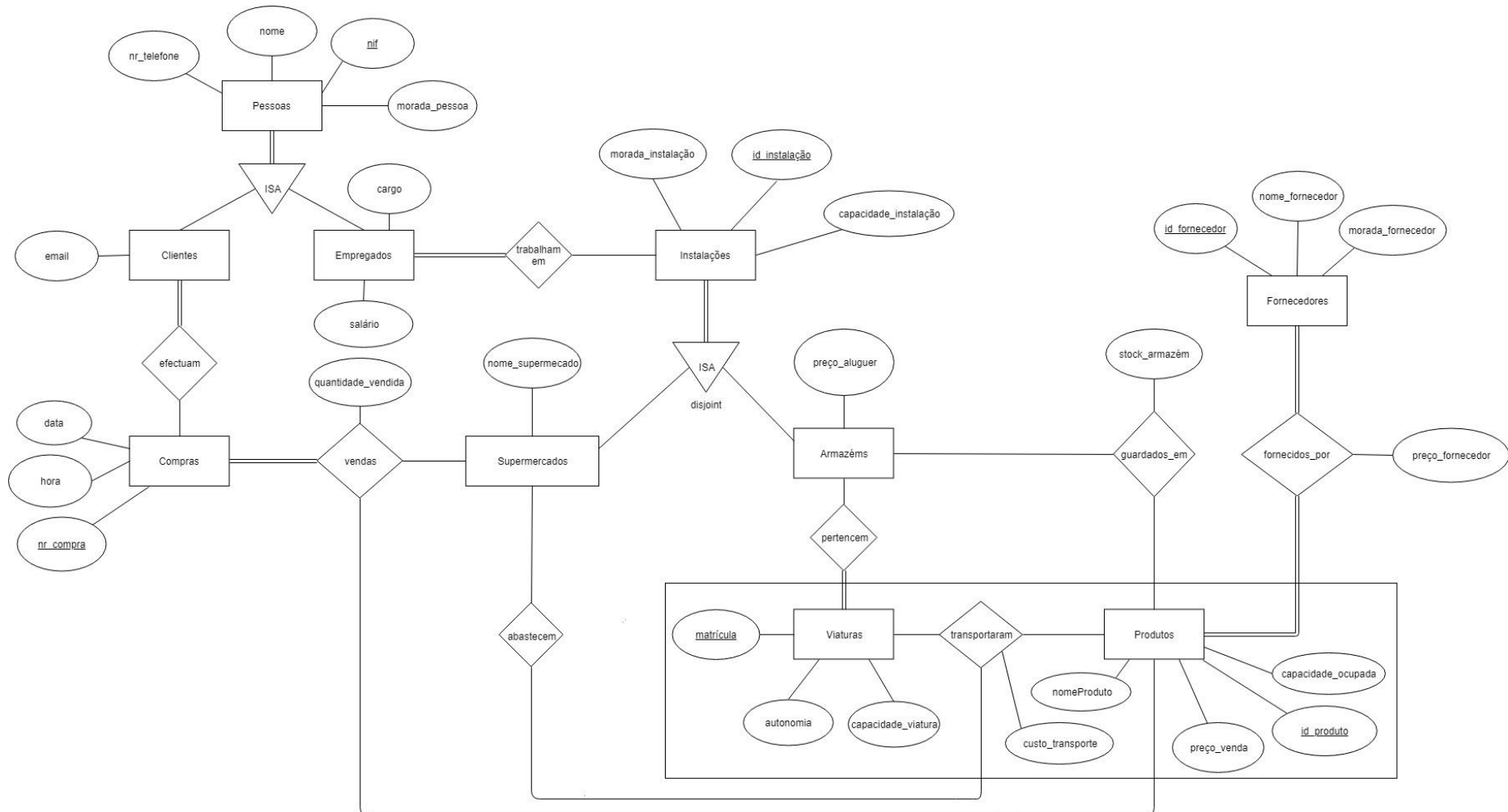
Na descrição inicial do trabalho falou-se em representar uma entidade Encomenda para representar os vários conjuntos de Produtos enviados pelo Fornecedor, mas entretanto decidimos simplificar o esquema e dizer que cada Produto é enviado por si só de cada vez, tendo um preço associado com cada Fornecedor. Isto permitiu-nos também adicionar as entidades Cliente e Compra, que achamos serem mais interessantes no contexto do nosso SGBD.

Cada Supermercado é abastecido por um ou mais Armazéns, através das suas Viaturas, que transportam os Produtos do Armazém para o Supermercado. Nos Armazéns são guardados os Produtos comprados aos Fornecedores que ainda não se encontram disponíveis para venda (ainda não foi vendido o stock presente no Supermercado).

Uma Viatura pertence a um determinado Armazém e tem associada uma matrícula e uma capacidade máxima de Produtos que podem transportar. As Viaturas servem para abastecer os Supermercados quando estes ficam sem stock ou com pouco stock de um ou mais Produtos. O transporte de Produtos por uma Viatura para um Supermercado tem associado um custo de transporte.

A cada Compra feita por um determinado Cliente ficam associados os produtos que a compõem, a quantidade que foi comprada de cada Produto e o Supermercado onde foi feita a Compra, actualizando-se após cada compra o stock de Produto presente no Supermercado onde foi realizada a Compra.

Modelo ER



Diogo Pereira, nº 44640; Eduardo Graça, nº 46794; João Vieira, nº 53332
27 de Maio de 2018

Esquema ER

As chaves primárias de cada relação são indicadas pelos atributos sublinhados.
Atributos a negrito foram adicionados/modificados da Fase 1.

pessoas(nif, nome, nr_telefone, morada_pessoa)

empregados(nif, cargo, salário) *nif é chave estrangeira de Pessoas*

clientes(nif, email) *nif é chave estrangeira de Pessoas*

fornecedores(id_fornecedor, nome_fornecedor, morada_fornecedor)

viaturas(matrícula, capacidade_viatura, autonomia)

instalações(id_instalação, morada_instalação, capacidade_instalação)

produtos(id_produto, preço_venda, capacidade_ocupada, **nome**)

armazéns(id_instalação, preço_aluguer) *id_instalação é chave estrangeira de Instalações*

supermercados(id_instalação, nome_supermercado) *id_instalação é chave estrangeira de Instalações*

compras(nr_compra, data, hora)

trabalham_em (nif, id_instalação) *nif é chave estrangeira de Pessoas e id_instalação é chave estrangeira de Instalações*

transportaram(matrícula, id_produto, custo_transporte) *matrícula é chave estrangeira de Viaturas e id_produto é chave estrangeira de Produtos*

abastecem(matrícula, id_produto, id_instalação) *matrícula é chave estrangeira de Viaturas, id_produto é chave estrangeira de Produto e id_instalação é chave estrangeira de Instalações*

vendas(id_produto, id_instalação, nr_compra, **quantidade_vendida**) *id_produto é chave estrangeira de Produtos, id_instalação é chave estrangeira de Instalações e nr_compra é chave estrangeira de Compras*

guardados_em(id_produto, id_instalação, stock_armazém) *id_produto é chave estrangeira de produto e id_instalação é chave estrangeira de Instalações*

fornecidos_por(id_fornecedor, id_produto, preço_fornecedor) *id_fornecedor é chave estrangeira de Fornecedores e id_produto é chave estrangeira de Produtos*

efectuam(nif, nr_compra) *nif é chave estrangeira de Pessoas e nr_compra é chave estrangeira de Compras*

pertencem(matrícula, id_instalações) *matrícula é chave estrangeira de Viaturas e id_instalações é chave estrangeira de Instalações*

Fase 2

Alterações ao Modelo ER

Em relação à primeira fase do trabalho, mudaram-se ligeiramente alguns atributos, nomeadamente modificando o atributo `stock_supermercado` na relação Vendas para `quantidade_vendida` e adicionando o atributo `nome` à entidade Produtos. Adicionalmente, devido a restrições na implementação do código SQL foi também adicionada uma entidade `Tipo_Instalação` que relaciona um código com um tipo de instalação (Supermercado ou Armazém). Decidimos não colocar esta entidade no modelo ER por não acharmos necessário a sua representação, pois esta apenas serve para a implementação da disjunção entre as duas entidades.

Discussão de Limitações e Opções Tomadas

Na criação da base de dados foi decidido criar como já referido anteriormente uma entidade `Tipo_Instalação` de forma a podermos implementar com sucesso a disjunção ISA indicada no esquema ER.

Criamos também algumas sequences de forma a que os identificadores das diversas tabelas fossem gerados automaticamente e assim na inserção dos dados iniciais das diversas tabelas fazemos uso de triggers e do comando `“currval”` para esse efeito em vez de inserir manualmente os valores e iniciar as sequences a partir do último valor adicionado.

Devido às restrições impostas na inserção de dados na tabela das Instalações, optámos por mostrar inserções, actualizações e remoções na tabela Produtos.

Triggers Criados

Para a realização deste trabalho, foram criados múltiplos triggers de sequência para as tabelas da base de dados que populam automaticamente o atributo da chave primária das seguintes entidades: Produtos, Fornecedores, Instalações e Compras. Adicionalmente, foram também adicionados triggers de verificação de constraints na inserção de dados nas relações Vendas e GuardadosEm e nas entidades Clientes e Empregados.

Triggers – Código SQL

```
/* Trigger para inserir um valor na tabela compras */
CREATE OR REPLACE trigger BI_compras
before insert on compras
for each row
begin
if :NEW.nrCompra is null then
select nrCompra_seq.nextval into :NEW.nrCompra from dual;
end if;
end;
/

/* Trigger para inserir um valor na tabela instalacoes */
CREATE OR REPLACE trigger BI_instalacoes
before insert on instalacoes
for each row
begin
if :NEW.idInstalacao is null then
select idInstalacao_seq.nextval into :NEW.idInstalacao from dual;
end if;
end;
/

* Trigger para inserir um valor na tabela produtos */
```

```
CREATE OR REPLACE trigger BI_produtos
```

```
before insert on produtos
```

```
for each row
```

```
begin
```

```
if :NEW.idProduto is null then
```

```
select idProduto_seq.nextval into :NEW.idProduto from dual;
```

```
end if;
```

```
end;
```

```
/
```

```
/* Trigger para inserir um valor na tabela fornecedores */
```

```
CREATE OR REPLACE trigger BI_fornecedores
```

```
before insert on fornecedores
```

```
for each row
```

```
begin
```

```
if :NEW.idFornecedor is null then
```

```
select idFornecedor_seq.nextval into :NEW.idFornecedor from dual;
```

```
end if;
```

```
end;
```

```
/
```

```
/* Trigger para verificar se a pessoa correspondente a este cliente já existe */
```

```
CREATE OR REPLACE trigger existeCliente
```

```
before insert on clientes
```

```
for each row
```

```
declare
```

```
p int;
```

```
begin
```

```
SELECT count(nif) into p
```

```
FROM Pessoas p
```

```
WHERE p.nif = :new.nif;
```

```
IF p = 0 then
```

```
Raise_Application_Error(-20001, 'A pessoa não existe! Insira primeiro em pessoas.');
```

```
end if;
```

```
end;
```

```
/
```

```
/* Trigger para verificar se a pessoa correspondente a este empregado já existe */
```

```
CREATE OR REPLACE trigger existeEmpregado
```



```

before insert on empregados
for each row
declare
p int;
begin
    SELECT count(nif) into p
    FROM Pessoas p
    WHERE p.nif = :new.nif;
    IF p = 0 then
        Raise_Application_Error(-20001, 'A pessoa não existe! Insira primeiro em pessoas.');
```

```

end if;
end;
/

--Triggers da aplicacao
CREATE OR REPLACE trigger adicionarStock
    before insert on guardadosEm
    for each row
declare
    capacidade number;
    ocupado number;
BEGIN
    SELECT i.capacidadeInstalacao into capacidade
    FROM Instalacoes i
    WHERE i.idInstalacao = :new.idInstalacao;
    SELECT sum(stockArmazem) INTO ocupado
    FROM guardadosEm;

    if :new.stockArmazem > capacidade + ocupado then
        Raise_Application_Error(-20001, 'Nao ha espaco para guardar todos os
produtos!');
    end if;
END;
/
```

```

CREATE OR REPLACE TRIGGER temProdutoParaVender
    before INSERT ON vendas
```

```
FOR EACH ROW
DECLARE
    quantidade number;
BEGIN
    SELECT sum(stockArmazem) INTO quantidade
    FROM guardadosEm
    WHERE idProduto = :new.idProduto;

    if :new.quantidadeVendida > quantidade then
        Raise_Application_Error(-20000, 'Não temos essa quantidade!');
    END if;
END;
/
```

Vistas Criadas

A vista que achámos mais interessante de implementar no nosso trabalho foi uma vista que nos apresentasse as despesas da Cadeia de Supermercados, o lucro das vendas e o lucro total. Para tal necessitámos de criar várias vistas auxiliares para obter os dados necessários e no fim um snapshot para materializar os dados.

A desvantagem disto é que não fica uma vista dinâmica, sendo necessário dar drop e criar novamente o snapshot para actualizar os dados, mas foi a solução que conseguimos implementar.

Vistas – Código SQL

```
drop view gastosSalariais;
```

```
create view gastosSalariais as  
select sum(salario) as salarios from empregados;
```

```
drop view gastosAluguer;
```

```
create view gastosAluguer as  
select sum(precoAluguer) as gastosAluguer from armazens;
```

```
drop view gastosProduto;
```

```
create view gastosProduto as  
select precoFornecedor, idProduto from fornecidosPor;
```

```
drop view precoProduto;
```

```
create view precoProduto as  
select idProduto, precoVenda from produtos;
```

```
drop view vendasProduto;
```

```
create view vendasProduto as  
select p.idProduto, (p.precoVenda * v.quantidadeVendida) as vendasProd from vendas v,  
precoProduto p  
where p.idProduto = v.idProduto;
```

```
drop view ganhosProduto;
```

```
create view ganhosProduto as  
select g.idProduto, (v.vendasProd - g.precoFornecedor) as ganhos  
FROM vendasProduto v, gastosProduto g  
WHERE v.idProduto = g.idProduto;
```

```
drop view ganhosVendas;
```

```
create view ganhosVendas as  
select sum(ganhos) as ganhosVendas  
from ganhosProduto;
```

```
drop snapshot lucroTotal;
```

```
create snapshot lucroTotal as  
select gs.salarios as gastosSalariais,  
       ga.gastosAluguer as gastosAluguer,  
       gv.ganhosVendas as ganhosVendas,  
       (gv.ganhosVendas - gs.salarios - ga.gastosAluguer) as lucroFinal  
from gastosSalariais gs, gastosAluguer ga, ganhosVendas gv;
```


Interface e Implementação da Aplicação

A aplicação desenvolvida em Apex pelo nosso grupo permite realizar quase todas as operações pedidas neste trabalho. Ao iniciar a aplicação é apresentada uma homepage com ligações para todas as outras páginas. Estas possuem todas o breadcrumb requisitado. Ao executar a aplicação, é possível:

- Visualizar os dados em tabelas (Páginas Instalações, Armazéns, Supermercados);
- Inserir, remover e actualizar tuplos da Base de Dados (Página Produtos);
- Listar dados onde códigos referentes a chaves externas são substituídos por outros atributos de fácil compreensão (Comando Empregados com Chave Substituída);
- Ver dados onde são apresentados valores derivados (Comando Empregados com Valores Derivados);
- Apresentar dois relatórios interligados em que um apresenta detalhes do outro (drill-down) (Comando Fornecedores com Drill-Down);
- Mostrar um detalhe condicional (Comando Fornecidos com detalhe condicional);
- Não implementados: Form Master-Detail e preenchimento de relações através de uma LOV.

Sequência de Operações

1. Iniciar a aplicação com o username UGBD29 e a password supermercado;

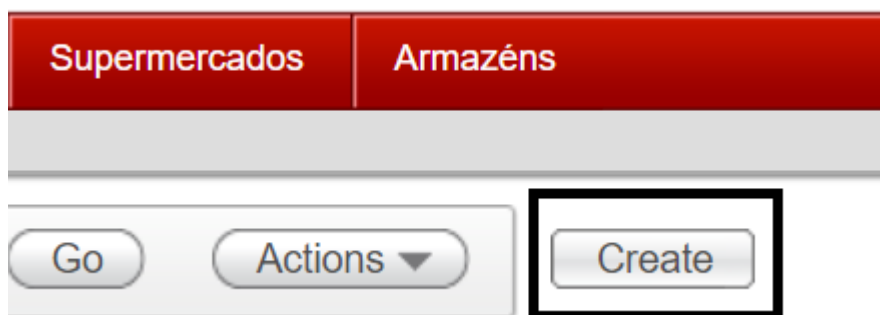
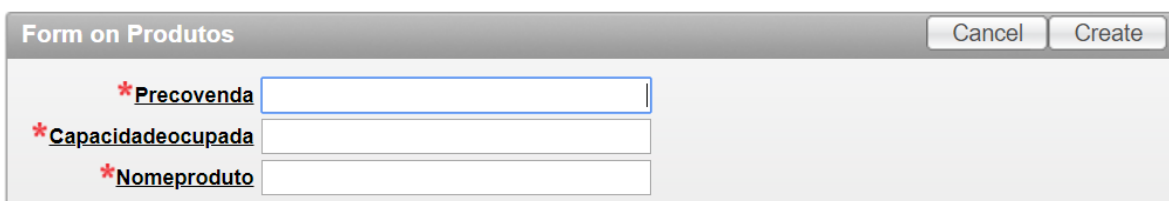


A login form titled "Login". It contains two input fields: "Username" with the value "ugbd29" and "Password" with masked characters ".....". A "Login" button is located to the right of the password field.

2. Navegar para uma das tabs Instalações, Armazéns ou Supermercados para verificar que a Base de Dados se encontra populada.



3. Navegar para a tab Produtos e realizar as inserções, remoções ou actualizações de Produtos desejadas.

A form titled "Form on Produtos" with "Cancel" and "Create" buttons at the top right. It contains three required fields, each marked with a red asterisk: "Precovenda", "Capacidadeocupada", and "Nomeproduto". Each field has an empty input box next to it.

	<u>Idproduto</u>	<u>Precovenda</u>	<u>Capacidadeocupada</u>	<u>Nomeproduto</u>
	300	\$1.00	12	agua
	301	\$3.99	10	laranja

Form on Produtos Cancel Delete Apply Changes

*Precovenda

*Capacidadeocupada

*Nomeproduto

4. Regressar à página inicial. Realizar cada um dos comandos desejados a partir da lista de comandos presente nesta.

Lista de Comandos

- Empregados com substituição de chave
- Produtos Fornecidos com detalhe condicional
- Fornecedores com drill-down
- Empregados com valor derivado
- Histórico de Vendas
- Lucros Cadeia de Supermercados

5. Para cada um dos comandos apresentados, bem como para os passos 1 a 3, verificar a existência da breadcrumb pedida.

Home **Produtos** **Instalações** **Supermercados** **Armazéns**

Home > Report on Produtos

 Go Actions ▼ Create